CS610 Assignment 4
Name: Manish (190477)

SYSTEM CONFIGURATION:

Architecture: x86_64
CPU op-mode(s): 32-bit, 64-bit
Byte Order: Little Endian
Address sizes: 39 bits physical, 48 bits virtual
CPU(s): 8
On-line CPU(s) list: 0-7
Thread(s) per core: 2
Core(s) per socket: 4
Socket(s): 1
NUMA node(s): 1
Vendor ID: GenuineIntel
CPU family: 6
Model: 142
Model name: Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz
Stepping: 12
CPU MHz: 2000.000
CPU max MHz: 4600.0000
CPU min MHz: 400.0000
BogoMIPS: 3999.93
Virtualization: VT-x
L1d cache: 128 KiB
L1i cache: 128 KiB
L2 cache: 1 MiB
L3 cache: 8 MiB
NUMA node0 CPU(s): 0-7

Compilation Details:

run.sh contains compilation commands for all 4 problems.

Problem 1:

Omp V1: Only uses tasks, No cutoff to serial code. For Omp V1 N=40 is assigned and compared with serial as it is taking lot of time for N=50. For rest of them N=50 is assigned and compared to serial code.
Omp V2: implemented using omp tasks, Cutoff to serial code. Three versions are tested when cutoff = 20, 25, 30.

Version	Cut-Off (Switch to serial version)	Speed Up
Omp V1	-	0.005x
Omp V2	20	1.91x
Omp V2	25	1.92x
Omp V2	30	**1.97x**
TBB (Blocking)	20	2.83x
TBB (Blocking)	25	2.84x
TBB (Blocking)	30	**2.84x**
TBB (continuation passing)	20	2.80x
TBB (continuation passing)	25	2.83x
TBB (continuation passing)	30	**2.84x**

We got the best speed up when cutoff=30 for all three Omp V2, TBB blocking style, and TBB continuation passing.
Serial code is very slow due to a lot of scheduling overheads because a lot of tasks are created near base cases.
Omp V2 runs faster because of low scheduling overheads as we switch to serial code for n less than the cutoff.

Problem 2:

Problem	Cut-Off (Switch to serial version)	Speed Up
Quicksort	50	3.22x
Quicksort	100	3.98x
Quicksort	300	3.77x
Quicksort	500	3.90x
Quicksort	700	4.13x
Quicksort	1000	**4.22x**
Quicksort	3000	3.58x
Quicksort	5000	3.56x
Quicksort	10000	3.15x

For Quicksort we switch to serial code for n less than the cutoff to avoid a lot of scheduling overheads because of lots of task creation near base cases.

Problem 3:

Vesion	Speed Up
Omp	3.61x
TBB	5.26x

OMP and TBB code has no issue of false sharing as I use reduction in both cases.

Problem 4:

Speed Up : **1.8x**
I used TBB parallel reduction to get speed up.

REFERENCES:

Class slides
INTEL TBB documentation shared on course website